Linear Algebra and its Applications 434 (2011) 2176–2186

Contents lists available at ScienceDirect

Linear Algebra and its Applications

journal homepage: www.elsevier.com/locate/laa

Syzygies for Metropolis base chains

Ian H. Dinwoodie

Fariborz Maseeh Department of Mathematics and Statistics, Portland State University, PO Box 751, Portland, OR 97207-0751, USA

ARTICLE INFO

Article history:
Received 29 March 2010
Accepted 16 December 2010
Available online 22 January 2011

Submitted by H. Schneider

AMS classification:
Primary: 13P10
60J10

Keywords:
Annealing
Metropolis algorithm
Sampling
Syzygies

ABSTRACT

It is shown that Markov chains for sampling from combinatorial sets in the form of experimental designs can be made more efficient by using syzygies on gradient vectors. Examples are presented.

1. Introduction

Designing base chains for use with the Metropolis algorithm is a problem with few standard techniques. Liu [14] says that "it is commonly agreed that finding an ideal proposal chain is an art" and Otten and Van Ginneken [16] say that the "quality – or lack of quality – of the chosen move set is not always that obvious." This paper is about an algebraic method to help design a good proposal chain.

There are at least two types of problems where the Metropolis algorithm is widely used: the first is simulating from a multidimensional distribution with a Markovian dependence structure, as in some physical models and Bayesian posterior calculations, and the second is for sampling approximately from a constraint set using the "annealing" version of the Metropolis algorithm in which the target distribution is concentrated on the set of interest. The first type of problem is presented in Liu [14], whereas here we focus on the second. Motivating examples are sampling from complex exponential

E-mail address: ihd@pdx.edu

0024-3795/$ - see front matter
doi:10.1016/j.laa.2010.12.021

models for networks such as social networks [19] and sampling from states in biochemical networks subject to constraints [12,6]. These examples are related to the integer-programming type of problem considered in Diaconis and Sturmfels [5], but they involve nonlinearity in the constraints. Sampling restricted permutations [4] can be put in this framework and will be considered in examples but the sampling method is not recommended for efficiency.

This paper relates closely to [6] where the goal was to sample from constrained discrete sets that correspond to objects like graphs and tables. There are examples where the sequential methods of that paper are not effective, in particular where the defining constraint functions are highly coupled (many variables are common to several constraint equations, making computational algebra difficult). The present work helps with those problems. It is not clear at this time how the computation time and memory requirements relate to other geometric notions of complexity (see [15]) and that would be an interesting research direction.

We argue, mostly with examples but also with some incomplete theory, that a good base chain should include proposals that are tangential to the constraint set, then we show how syzygies can be used to get a rich set of tangential moves that are state-dependent for a symmetric proposal kernel. Use of the derivative of a function defining a target exponential distribution has appeared in Hanson [11], and increments in random directions is formalized as the "random-grid method" in Liu [14], and both ideas appear here. The method of syzygies, while conceived for nonlinear constraints, can be applied to linear constraints and in that case produces increments (moves) for a Markov chain that are close to but possibly weaker than a "lattice basis," in the terminology of the literature of Markov chains on polytopes.

2. State dependent increment sets

For each $i = 1, \ldots, c$, let $g_i : \mathbb{R}^d \to \mathbb{R}$ be a polynomial function with integer coefficients. Assume $c \leq d$ and define the set Ω by

$$\Omega := L \cap \bigcap_{i=1}^{c} \{g_i(\mathbf{x}) = 0\}$$

which is a *fractional design* as a subset of the product space $L := \{0, 1, 2, \ldots, l-1\}^d$ [17,18]. Points in L will be written as column d-tuples like $\mathbf{x}$ and $\mathbf{y}$. The level l may be 2 in certain cases, but in biological network applications its value comes from a discretization step and is often 3 or 4 [12].

Let

$$\pi_\theta(\mathbf{x}) = \frac{e^{-\theta U(\mathbf{x})}}{z_\theta}, \ \mathbf{x} \in L \tag{1}$$

where $U := -\sum_{i=1}^{c} g_i^2$. Sampling from L with probability distribution π_θ will generate points with high probability in Ω, and with a conditionally uniform distribution on Ω, when θ is large. Approximate sampling can be done with the Metropolis algorithm [14] using any symmetric, irreducible proposal kernel K on L – run a Markov chain in L with transition matrix K_θ for stationary distribution π_θ and wait for it to converge. The standard method is

$$K_\theta(\mathbf{x}, \mathbf{y}) = K(\mathbf{x}, \mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}.$$

Some proposal kernels K will be more efficient than others, in that the proportion of rejected proposal moves will be smaller, leading to more mobility in the state space, faster convergence to stationarity, and ultimately more accurate estimates of expectations using time averages.

Let R be the ring of polynomials $\mathbb{Q}[\mathbf{s}] = \mathbb{Q}[s_1, \ldots, s_d]$. Define the gradient $\nabla g_i = (\partial_j g_i)_{j=1,\ldots,d} \in R^d$. Let

$$\partial_j G := \begin{pmatrix} \partial_j g_1 \\ \cdot\cdot \\ \cdot\cdot \\ \partial_j g_c \end{pmatrix}$$

and define the module J to be the span of the polynomial c-tuples $\partial_j G$, with polynomial coefficients $f_j \in R$:

$$J := \left\{ \sum_{j=1}^{d} f_j \cdot \partial_j G \right\} \in R^c.$$

Consider the syzygy module $S_J \subset R^d$ of d-tuples on the generators $\partial_1 G, \ldots, \partial_d G$ defined by

$$S_J := \{(p_1, \ldots, p_d) \in R^d : p_1 \cdot \partial_1 G + p_2 \cdot \partial_2 G + \cdots + p_d \cdot \partial_d G = 0\}.$$

This can be written in the form

$$\nabla G \cdot P = 0$$

if $P = (p_1, \ldots, p_d)$ is the column of polynomials and G is the derivative matrix

$$\nabla G := \begin{pmatrix} \partial_1 G & \ldots & \partial_d G \end{pmatrix} = \begin{pmatrix} \nabla g_1 \\ \cdots \\ \cdots \\ \nabla g_c \end{pmatrix}.$$

Assume that G is of full rank c, in the sense that the row vectors with polynomial entries are linearly independent over the field of rational functions. Now let M_{S_J} be a $d \times g$ matrix of generators (as columns) for S_J, that is a matrix whose columns are $d \times 1$ vectors of polynomials that are in the module S_J and whose span (with polynomial coefficients) is all of S_J. This matrix M_{S_J} is called a presentation matrix for the module J [3]. Write

$$M_{S_J} := \begin{pmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_g \end{pmatrix}$$

for the $d \times g$ generating matrix of syzygies.

Each column $\mathbf{v}$ of the matrix M_{S_J} evaluated at the point $\mathbf{x} \in \Omega$ satisfies

$$\nabla g_i(\mathbf{x}) \cdot \mathbf{v}(\mathbf{x}) = 0, \; i = 1, \ldots, c$$

and therefore is tangent to each constraint function at $\mathbf{x}$. This statement follows from the definitions. Then observe that the acceptance probability $e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}$ in the kernel $K_\theta(\mathbf{x}, \mathbf{y}) := K(\mathbf{x}, \mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}$ will be on the order of $e^{-\theta\lambda^\star \|\mathbf{y}-\mathbf{x}\|^2/2}$ if $\mathbf{y} = \mathbf{x} \pm \mathbf{v}_i(\mathbf{x})$, where $\lambda^\star$ is the spectral radius of the second derivative of U at $\mathbf{x}$. This follows from a Taylor expansion and the simple fact that $\nabla U(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) = 0$. Since our state space is in the integers, $\|\mathbf{y} - \mathbf{x}\|$ is not necessarily small, but nevertheless increments that satisfy $\nabla g_i(\mathbf{x}) \cdot \mathbf{v}(\mathbf{x}) = 0$ should lead to lower rejection rates and better mobility in the state space L.

Proposition 2.1 below says that the span of the columns is essentially all of the nullspace of ∇G at each point $\mathbf{x}$, except for some points of degeneracy.

Proposition 2.1. *Let $\mathbf{x} \in L$ be a particular point, and let a point $\mathbf{y} \in L$ satisfy $\nabla G(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) = 0$. If the matrix $\nabla G(\mathbf{x})$ is of full rank and if the matrix $M_{S_J}(\mathbf{x})$ is of full rank, then $\mathbf{y}$ can be represented as*

$$\mathbf{y} = \mathbf{x} + P(\mathbf{x})$$

for some syzygy $P = (p_1, \ldots, p_d) \in S_J$.

Proof. By assumption, the increment $\mathbf{y} - \mathbf{x}$ is in the kernel of $\nabla G(\mathbf{x})$, and since $\nabla G(\mathbf{x})$ is of full rank, the dimension of the nullspace of $\nabla G(\mathbf{x})$ is $d - c$.

Now the matrix ∇G over the rational functions (polynomial fractions) also has full rank, and its null space (with respect to rational functions) is spanned by $d - c$ independent vectors of rational functions, which in fact may be taken to be polynomials by clearing denominators. Thus there exist $d - c$ independent polynomial vectors in the nullspace of ∇G and so the matrix of syzygies M_{S_J} has at least $d - c$ columns. Since we are assuming $M_{S_J}(\mathbf{x})$ is of full rank, that rank must be at least $d - c$, which is sufficient to span the kernel of $\nabla G(\mathbf{x})$. Thus we can write

$$\mathbf{y} - \mathbf{x} = M_{S_J}(\mathbf{x}) \cdot \mathbf{a}$$

for a vector $\mathbf{a}$ of rational coefficients on the columns. Then finally we set

$$P = M_{S_J} \cdot \mathbf{a}$$

which will evaluate to $\mathbf{y} - \mathbf{x}$ at $\mathbf{x}$. $\square$

Note that the coefficients $\mathbf{a}$ that give the linear combination of columns of M_{S_J} in the proof of Proposition 2.1 can be taken to be integers if $M_{S_J}(\mathbf{x})$ is totally unimodular.

The assumptions of full rank are hard to remove, as there are examples, such as Example 2.1 below, where the rank of the matrix of generators drops down at certain points. Such points are called "invariant zeros" in linear systems [9] and raise interesting technical issues. Constant rank in the presentation matrix M_{S_J} is related to whether J is free (see [3, Theorem 4.14]). See also Lin [13] for a discussion of syzygy modules of polynomial matrices with motivation in control theory and [7] for applications in coding and [10] for module definitions.

Our Markov chain on L will start with a basic irreducible transition kernel $B_s(\mathbf{x}, \mathbf{y})$ defined by choosing s sites randomly and uniformly from the d dimensions, and randomizing their values to $0, 1, \ldots, l - 1$ with equal probability. This part is necessary to guarantee irreducibility, since this property will not be guaranteed with the moves based on syzygies.

The obvious first choice for the Markov chain is to choose either to move with B_s with probability $1/2$ or alternatively with probability $1/2$ to choose one of the syzygies uniformly and add its value to the current state. While enhancing the basic kernel B_s, the nature of this simple combination of B_s and the syzygy moves, in terms of limiting distribution or symmetry as a base chain for the Metropolis algorithm, is not clear because of the complicated state-dependent nature of the syzygies. So we will define more carefully a Markov transition kernel K_S from the syzygies that will be symmetric, then combine it by a simple coin flip with the basic kernel B_s to guarantee irreducibility in our Markov kernel K. In fact any positive weights on the two kernels will give a valid symmetric and irreducible chain K, but we will use weights 1/2 and 1/2 for simplicity.

Now let us define carefully the part K_S of the base, symmetric Markov chain K. Suppose, by way of motivation, that $K_S(\mathbf{x}, \mathbf{y})$ selected a column $\mathbf{v}$ of M_{S_J} uniformly, and added its randomly-signed evaluation $\sigma\mathbf{v}(\mathbf{x})$ to the current state $\mathbf{x}$. This procedure is not necessarily symmetric, since at $\mathbf{x}$ the columns are those in $M_{S_J}(\mathbf{x})$ whereas at $\mathbf{y}$ they are those in $M_{S_J}(\mathbf{y})$, so the probability of moving from $\mathbf{x}$ to $\mathbf{y}$ is not clearly the same as the probability of moving from $\mathbf{y}$ to $\mathbf{x}$. The Metropolis–Hastings algorithm could be used to correct this, but it requires finding the ratio of the probabilities of moving in the two directions, which would be difficult to compute. So a symmetrized version will be defined next with two steps: first a uniform selection of column of M_{S_J}, then an acceptance step that involves another random choice of column.

Recall that M_{S_J} is a $d \times g$ matrix with polynomial entries. Let $N_{\mathbf{x}}(\delta)$ denote the number of columns that evaluate to $\pm\delta$ at $\mathbf{x}$. Then we define

$$K_S(\mathbf{x}, \mathbf{y}) = \frac{N_{\mathbf{x}}(\mathbf{y} - \mathbf{x})}{2g} \cdot \frac{1}{N_{\mathbf{x}}(\mathbf{y} - \mathbf{x})} = \frac{1}{2g} \tag{2}$$

for $\mathbf{y} \neq \mathbf{x}$ when both $N_{\mathbf{x}}(\mathbf{y} - \mathbf{x}) > 0$, $N_{\mathbf{y}}(\mathbf{y} - \mathbf{x}) > 0$. The practical implementation is as follows: choose a column of $M_{S_J}(\mathbf{x})$ uniformly, let its value be δ, and randomize its sign to get $\sigma\delta$. Then (to symmetrize the kernel) select another column uniformly at random from the ones in $M_{S_J}(\mathbf{x})$ that are $\pm\delta$. If the second choice of column is the same as the first, and if $\mathbf{y} = \mathbf{x} + \sigma\delta \in L$ and if $\pm\delta$ is any column of $M_{S_J}(\mathbf{y})$, then accept the move to $\mathbf{y}$, otherwise hold. Note that if the matrix M_{S_J} has constant, linearly independent vectors, as would arise with linear constraints, then $N_{\mathbf{x}}(\delta) \in \{0, 1\}$ (at most one column can be $\pm\delta$), so the second acceptance step with probability $\frac{1}{N_{\mathbf{x}}(\mathbf{y}-\mathbf{x})} = 1$ will never be rejected. For some applications where exploring Ω is more important than precise probability calculations, it may not be necessary to symmetrize K_S.

Finally, define the symmetric, irreducible kernel to be

$$K = \frac{1}{2} B_s(\mathbf{x}, \mathbf{y}) + \frac{1}{2} K_S(\mathbf{x}, \mathbf{y}).$$

The symmetric kernel K can be used as a Metropolis base chain to sample approximately from π_θ in the usual way:

$$K_\theta(\mathbf{x}, \mathbf{y}) = K(\mathbf{x}, \mathbf{y}) \cdot \min\{1, e^{-\theta(U(\mathbf{y}) - U(\mathbf{x}))}\}.$$

Example 2.1. Consider the set Ω to be the symmetric graphs on 4 vertices, with 4 edges and 1 triangle. The adjacency matrices are a subset of binary sequences of length 6, and are written

$$X = \begin{pmatrix} 0 & & & \\ x_1 & 0 & & \\ x_2 & x_3 & 0 & \\ x_4 & x_5 & x_6 & 0 \end{pmatrix}.$$

We will abuse notation in this example and others and let the variables x_j be both indeterminates (officially s_j) and points in L. The matrix ∇G is 2×6 with a row of ones for the edge count, and a row of partial derivatives of the polynomial that counts triangles (which can be written as the trace of the adjacency matrix cubed, divided by 6):

$$\nabla G = \begin{pmatrix} 1 & \cdots & 1 \\ x_2x_3 + x_4x_5 & \cdots & x_2x_4 + x_3x_5 \end{pmatrix}.$$

The probability π_θ from (1) will have very nearly 99% of its mass on Ω if $\theta = 4$. A standard base chain $B_1(\mathbf{x}, \mathbf{y})$ is one where a vertex pair $\{i, j\}$ is chosen uniformly and its edge status is switched. That is, $s = 1$ in the previous notation and one "site" is chosen from $d = \binom{4}{2}$ binary coordinates coding edge presence. We argue now that the Metropolis algorithm with proposal kernel B_1 and no K_S in the definition of K is practically immobile at this value of θ, but bringing in the syzygies improves the chain dramatically.

Consider the graph $\mathbf{x}$ with the triangle on $1, 2, 3$ and fourth edge from vertex 3 to 4, so edge set $\{1, 2\}, \{1, 3\}, \{2, 3\}, \{3, 4\}$ Observe that for the chain with kernel B_1 to move to the graph $\mathbf{y}$ with edge set $\{1, 3\}, \{1, 4\}, \{2, 3\}, \{3, 4\}$ (triangle on $1, 3, 4$), there has to be an edge switch which changes simultaneously both the edge count and the triangle count away from their target values 4 and 1. Then the acceptance probability $\min\{1, e^{-\theta(U(\mathbf{y})-U(\mathbf{x}))}\}$ will be no more than $e^{-\theta\cdot 2} = .00034$. This implies that the Metropolis chain based on the kernel B_1 will hold at some state between the first graph and the second on average nearly 3000 steps.

Singular [8] gives a set of 11 generators using graded reverse lex order for the syzygies on the Jacobean J. For example, the first one is the column vector

$$(0, -x_2 + x_5, x_3 - x_4, -x_3 + x_4, x_2 - x_5, 0).$$

Consider the state $\mathbf{x}$ which has a triangle on vertices 1,2,3, and edge connecting vertex 4 to 3. This is one of 12 states, that in this case can be obtained from one another by permuting vertex labels. The syzygies evaluate to vectors in a matrix, with rows indexed by edges for variables $x_1, \ldots, x_6$:

$$\begin{array}{c} \\ x_1, 21 \\ x_2, 31 \\ x_3, 32 \\ x_4, 41 \\ x_5, 42 \\ x_6, 43 \end{array} \begin{array}{c} \begin{array}{ccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & 0 & -1 & 1 & -1 & 0 & 2 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & -1 & 1 & 0 & -2 \\ -1 & -1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Here we see that column 10 added to the present graph will remove edge $\{1, 2\}$ and add edge $\{1, 4\}$, taking us directly from $\mathbf{x}$ to $\mathbf{y}$.

Example 2.2. Consider restricted permutations, first on three characters for definitions and notation. A permutation with no fixed points can be written as a 3×3 0-1 matrix $A = (a_{ij})$ with exactly one 1 in each row and column, and zeros on the diagonal. These can be written as 9 polynomial constraints – 6 for the row and column sums and three for the forbidden diagonals. The polynomials are linear and lead to the derivative matrix ∇G with columns indexed in the order $a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, a_{31}, a_{32}, a_{33}$:

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

The matrix ∇G has rank 8, and its kernel is spanned by the syzygy written back in permutation matrix form as the increment

$$\delta = \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix}.$$

This single increment will cycle between the two feasible permutations, and clearly a Markov chain for the Metropolis algorithm which changes one or two entries will be practically frozen. This derangement example is considered in Remark 3.5 of Diaconis et al. [4] as an example where transpositions are not sufficient for connectivity, and the syzygy method gets this correct. On the $3n \times 3n$ examples that build on this one presented in that paper, the syzygies give some transpositions and some more complicated steps like the one above.

More realistic dimensions for this type of problem involve permuting around 100 characters. An example in Diaconis et al. [4] has a 0-1 matrix coding permutations of size 210×210 from a problem in astronomy. The restrictions come from truncation and require that the 1 in each row be on the left of a bound for that row (that is, in an interval of integers [1, b] possibly strictly inside [1,210]). This is a "one-sided" restriction problem, and there is an efficient sampling method. For us, this problem requires $210^2 = 44{,}100$ indeterminates, which is not theoretically impossible in symbolic software but Singular limits the number to 32,767. Thus we consider a simplified 100×100 problem, with restriction that the final entry [100,100] be 0 (the value of the permutation at 100 cannot be 100) as a test of computational feasibility. With the help of the `genmodel` feature of 4ti2 [1], we can easily construct the matrix ∇G as the derivative of linear constraints. The syzygy computation essentially constructs a nullspace for the matrix, but in a way that is more useful than standard numerical methods of linear algebra which give orthonormal bases that have little connection to the integer state space. Singular very quickly computes a basis (free generators) of 9800 syzygies (the dimension of the nullspace of $\nabla G(\mathbf{x})$), which are transpositions. 4ti2 also gives a lattice basis of 9800 increments. In the end, a method based on random increments to a 0-1 table representation of a permutation will not be efficient, as too many of the increments will give inadmissible tables, but the method gives insight into the complexity of changes that must be made to preserve constraints.

When there is one constraint $g_1(\mathbf{x}) = 0$ in a certain form, the syzygies can be counted easily, but in general knowing the number of generators of S is hard.

Proposition 2.2. *Let there be one constraint $g_1(\mathbf{x}) = 0$, and suppose there is a variable x_α such that $g_1(\mathbf{x}) = x_\alpha + f(\mathbf{x})$ where $f(\mathbf{x})$ does not depend on x_α. Then the syzygies are a free module with basis of size $d - 1$.*

Proof. Here the matrix ∇G has one row, and one entry is 1. This means by the Quillen–Suslin Theorem [3, p. 187, see also p. 194] that the module S is free. In fact a basis is given explicitly by $\mathbf{v}_j = \partial_j g_1 \mathbf{e}_1 - \mathbf{e}_j$, $j = 2, \ldots, d$, where $\mathbf{e}_j$ is the standard basis element of R^d, when $\alpha = 1$ for concreteness. □

3. Computation and approximation

For some realistic simulation problems in statistics, it will not be possible to compute the presentation matrix M_{S_J} that gives a generating set for the syzygies. The worst case analysis for computation of syzygies is bad [2]. There are two approaches then to deal with complex problems where full computation is not possible: (1) use a subset of the syzygies for each constraint computed separately, and intersect the corresponding modules, or (2) use an approximation based on circuit substitution.

To explain the first method, let S_i be the syzygy module in R^d for the d-tuple ∇g_i:

$$S_i := \{(p_1, \ldots, p_d) : \sum_{j=1}^{d} p_j \cdot \partial_j g_i = 0\}, \; i = 1, \ldots, c.$$

Proposition 3.1. *If $S_i \supset T_i$, $i = 1, \ldots c$, then $S_J \supset T_1 \cap T_2 \cap \cdots \cap T_c$.*

Proof. Nearly by definition, $S_J = S_1 \cap S_2 \cdots \cap S_c$, then the result follows. □

Example 3.1. Consider symmetric graphs on 5 vertices, with 2 triangles and 6 edges. Unlike Example 2.1, here not all the unlabelled graphs are topologically equivalent and the computations are harder. Computing S_J is difficult, but separately S_E and S_T, the syzygy modules for the edge and triangle functions, are easy. The intersection

$$S_E \cap S_T[1 \ldots k]$$

of S_E with the module of the first k generators of S_T can be computed easily up to approximately 16 of the 83 minimal generators of S_T.

The second method for cheap syzygies is based on the circuit polynomials [20]. Recall that ∇G is a $c \times d$ matrix and let $c < d$. Consider $c \times d$ indeterminates y_{ij} in a matrix Y:

$$Y = \begin{pmatrix} y_{11} & \cdots & \cdots & y_{1d} \\ \cdots & \cdots & \cdots & \cdots \\ y_{c1} & \cdots & \cdots & y_{cd} \end{pmatrix}.$$

For each subset $C = \{\tau_1, \ldots, \tau_{c+1}\}$ of the $\binom{d}{c+1}$ subsets of size $c+1$ of column indices, form the $d \times 1$ vector $\mathbf{v}_C$ with nonzero entries at coordinates τ_k given by:

$$\mathbf{v}_{C,\tau_k} := (-1)^k \det(Y_{C-\tau_k}),\ k = 1, \ldots, c+1 \tag{3}$$

where $Y_{C-\tau_k}$ is the matrix with only the c columns indexed by $C - \{\tau_k\}$. By Cramér's Rule, each vector $\mathbf{v}_C$ is in the kernel of Y with polynomial entries. Now substitute the polynomials $\partial_j g_i(\mathbf{s})$ in for y_{ij} and the result is a syzygy.

Proposition 3.2. *Let $\mathbf{v}_C(\mathbf{y})$ be the polynomial vector in indeterminates y_{ij} defined at (3) above, and let P_C be a d-tuple of polynomials given by $P_C = \mathbf{v}_C(\partial_j g_i(\mathbf{s}))$. Then $\nabla G \cdot P_C = 0$.*

Proof. For any value $\mathbf{x} \in \mathbb{Q}^d$, it holds that the rational vector

$$\nabla G(\mathbf{x}) \cdot P_C(\mathbf{x}) = Y \cdot \mathbf{v}_C/_{(y_{ij}=\partial_j g_i(\mathbf{x}))} = 0.$$

Then over the infinite field $\mathbb{Q}$ it follows that in indeterminates $\mathbf{s}$, we have $\nabla G(\mathbf{s}) \cdot P_C(\mathbf{s}) = 0$ which proves that P_C is the claimed syzygy. $\square$

In practice it is not always necessary to get the syzygies P_C from the formula (2). One can use algebra software for the syzygies on the matrix Y, a calculation which can be done fairly easily compared to ∇G despite the large number of indeterminates. Then substitute the polynomial entries $\partial_j g_i(\mathbf{s})$ for y_{ij} in the columns. Another useful observation for difficult computational problems is that each vector $\mathbf{v}_C$ has 0 entries except in the $c+1$ places marked by $\tau_1, \ldots, \tau_{c+1}$. Thus, on evaluation at a state space $\mathbf{x}$, at most $c+1$ will be nonzero, and this implies that a good choice of s in the kernel B_s (where s is the number of sites to update in the crudest base kernel) is $c+1$.

Example 3.2. Consider again the set of graphs with a fixed number of triangles and edges. With 4 vertices the matrix G is 2×6 and then the Y matrix is given by

$$Y = \begin{pmatrix} y_{11} & y_{12} & y_{13} & y_{14} & y_{15} & y_{16} \\ y_{21} & y_{22} & y_{23} & y_{24} & y_{25} & y_{26} \end{pmatrix}.$$

We can obtain the $\binom{6}{3} = 20$ circuit syzygies by taking any three columns, and solving for the last one in terms of the other two, using Cramér's Rule. Singular reports the following for example, corresponding to using the last three columns:

$$[0, 0, 0, y_{15}y_{26} - y_{16}y_{25}, -y_{14}y_{26} + y_{16}y_{24}, y_{14}y_{25} - y_{15}y_{24}].$$

Then substitute $y_{1j} = 1, j = 1, \ldots, 6$, and $y_{21} = x_2x_3 + x_4x_5$ using the indeterminates as in Example 2.1, etc.

For larger network problems, the circuit polynomials are convenient because they do not need to be stored. For example, suppose we want to simulate symmetric graphs on 32 nodes, inspired by the EIES data sets on 32 researchers at the Siena webpage `http://stat.gamma.rug.nl/`. The constraint for a fixed number of edges is linear, so the first row of the matrix ∇G is 1, and we can take the first row of Y to be 1 as well. Then the second row of Y has indeterminates $y_1, \ldots, y_{496}$. The circuit syzygy $\mathbf{v}_C$ for each set of three column indices $C = \{i, j, k\}$ (with $i < j < k$ these number $\binom{496}{3} = 20{,}214{,}480$, too many for practical storage) has a simple explicit formula:

$$\mathbf{v}_C(i) = y_j - y_k$$
$$\mathbf{v}_C(j) = y_k - y_i$$
$$\mathbf{v}_C(k) = y_i - y_j$$

with vanishing entries elsewhere. Then substitution $y_j = \partial_j p_T$ where p_T is the function that counts triangles is given by

$$\mathbf{y} = \nabla\ \mathrm{trace}(X^3)/3!$$

where X is the symmetric adjacency matrix used in Example 2.1. On 5 vertices, the formula for the 10-tuple $P_{\{1,2,3\}}$ is

$$P_{\{1,2,3\}} = \begin{pmatrix} -x_1x_2 + x_1x_3 + x_4x_6 - x_5x_6 + x_7x_9 - x_8x_9 \\ x_1x_2 - x_2x_3 - x_4x_5 + x_5x_6 - x_7x_8 + x_8x_9 \\ -x_1x_3 + x_2x_3 + x_4x_5 - x_4x_6 + x_7x_8 - x_7x_9 \\ 0 \\ \cdots \\ 0 \end{pmatrix}.$$

This evaluates to some moves that change two edges on certain graphs. Since the circuits will never change more than three edges when two constraints ($c = 2$) are in place, there are reasons to seek more complicated syzygies.

These network problems will not yield to full computation of generators M_{S_J}, but practical approximations such as the circuits should be useful.

Example 3.3. Consider 3-level polynomial dynamics given by

$$f_1(x_1, x_2, x_3) = 2x_2x_3 + 2x_2 + 2x_3$$
$$f_2(x_1, x_2, x_3) = 2x_3^3 + x_2^2 + x_2 + 2x_3$$
$$f_3(x_1, x_2, x_3) = 2x_3^2 + 2x_1 + 2$$

in L with $l = 3$ and modulo 3 operations and $d = 3$ coordinates (this is Example 10.3 in Laubenbacher and Stigler [12]). Let Ω be the preimage of state (a, b, c):

$$\Omega := \{\mathbf{x} : f_1(\mathbf{x}) = a, f_2(\mathbf{x}) = b, f_3(\mathbf{x}) = c\}.$$

Then using rules for calculus in the real number field we get

$$\nabla G = \begin{pmatrix} 0 & 2x_3 + 2 & 2x_2 + 2 \\ 0 & 2x_2 + 1 & 6x_3^2 + 2 \\ 2 & 0 & 4x_3 \end{pmatrix}.$$

The syzygy module $S_J = 0$, that is the generators of J are free (this is true over $\mathbb{Q}$ and over $\mathbb{Z}_3$), and are therefore not useful for sampling from Ω.

An example where the syzygies are useful (but where ∇G is not of full rank) is one where $\mathbf{x}=(x_1, x_2, x_3, x_4, x_5, x_6)$ (integers mod 3) and $f_i(\mathbf{x})=x_i \cdot x_{i+1}$ for indices $i = 1, \ldots, 6$, and index summation is circular, so $f_6(\mathbf{x})=x_6 \cdot x_1$. Then the syzygies on J are generated by one element $\mathbf{v}=(x_1, -x_2, x_3, -x_4, x_5, -x_6)$. Suppose Ω is the preimage of $\mathbf{0}$ (a steady state solution for $(f_1, f_2, f_3, f_4, f_5, f_6)$). The vector $\mathbf{v}$ will give a move from the point $\mathbf{x} = (1, 1, 1, 1, 1, 1) \in L$ to $\mathbf{y} = (2, 0, 2, 0, 2, 0) \in \Omega$ but not back, so this move is not allowed in the symmetric K_S. From $\mathbf{y}$ one can move to $(1, 0, 1, 0, 1, 0)$ and back with addition mod 3, a move that is not available with the one or two site updates B_1, B_2. Here it would be natural to factor out the gcd of the components of $\mathbf{v}$.

4. Conclusions

We have shown that it can be useful to compute syzygies on the columns of the derivative matrix ∇G when trying to sample from a discrete constrained set of the form $G(\mathbf{x}) = 0$. The syzygies give a set of tangent vectors that serve as good increments in a Metropolis base chain. While the tangent geometry is weak for discrete subsets, examples show that the method adds valuable moves to basic proposal kernels. There are strategies for approximating a generating set of syzygies that are useful when the full syzygy computation cannot be done, and current algebra software is capable of handling realistic problems with thousands of variables. Furthermore, examining the syzygies evaluated at points in the state space gives insight into the number of sites s to update in the basic proposal kernel B_s. The method adds little to current methods when the constraints are linear, but it does help with designing Markov chains with nonlinear constraints. Further work could be on improving the symmetrized kernel K_S at (2) to reduce the rejection rate, use of the unsymmetrized version, numerical studies on a range of social network models, and extensions to a continuous state space.

References

[1] 4ti2 Team, 4ti2—A Software Package for Algebraic, Geometric and Combinatorial Problems on Linear Spaces, 2009. Available from: <http://www.4ti2.de>.

[2] D. Bayer, M. Stillman, On the complexity of computing syzygies, J. Symbolic Comput. 6 (1988) 135–147.

[3] D. Cox, J. Little, D. O'Shea, Using Algebraic Geometry, Springer, New York, 1998.

[4] P. Diaconis, R. Graham, S.P. Holmes, Statistical problems involving permutations with restricted position, in: State of the Art in Probability and Statistics (Leiden, 1999), IMS Lecture Notes Monogr. Ser., 36, IMS, Beachwood, Ohio, 2001, pp. 195–222.

[5] P. Diaconis, B. Sturmfels, Algebraic algorithms for sampling from conditional distributions, Ann. Statist. 26 (1998) 363–397.

[6] I.H. Dinwoodie, Polynomials for classification trees and applications, Statist. Methods Appl. 19 (2010) 171–192.

[7] E. Fornasini, M.E. Valcher, nD polynomial matrices and applications to multidimensional signal analysis, Multidimens. Systems Signal Process. 8 (1997) 387–408.

[8] G.-M. Greuel, G. Pfister, H. Schönemann, Singular 3.1.0 A Computer Algebra System for Polynomial Computations. Centre for Computer Algebra, University of Kaiserslautern, 2009. Available from: <http://www.singular.uni-kl.de>.

[9] T. Kailath, Linear Systems, Prentice-Hall, Englewood Cliffs, 1980.

[10] M. Kreuzer, L. Robbiano, Computational Commutative Algebra I, Springer, New York, 2000.

[11] K. Hanson, Markov chain Monte Carlo posterior sampling with the Hamiltonian method, Proc. SPIE 4322 (2001) 456–467.

[12] R. Laubenbacher, B. Stigler, Design of experiments and biochemical network inference, in: P. Gibilisco (Ed.), Algebraic and Geometric Methods in Statistics, Cambridge University Press, New York, 2010, pp. 175–185.

[13] Z. Lin, On syzygy modules for polynomial matrices, Linear Algebra Appl. 298 (1999) 73–86.

[14] J.S. Liu, Monte Carlo Strategies in Scientific Computing, Springer, New York, 2001.

[15] H. Maruri, E. Sáenz-Cabezón, H.P. Wynn, Complexity of experimental designs and polynomial models via Betti Numbers, in: SAMSI Transition Workshop, June 19, 2009.

[16] R.H.J.M. Otten, L.P.P.P. Van Ginneken, The Annealing Algorithm, Kluwer Academic Publishers, Norwell, Massachusetts, 1989.

[17] G. Pistone, E. Riccomagno, H. Wynn, Algebraic Statistics: Computational Commutative Algebra in Statistics, Chapman and Hall, New York, 2001.
[18] G. Pistone, M.-P. Rogantin, Indicator function and complex coding for mixed fractional factorial designs, J. Statist. Plann. Inference 138 (2008) 787–802.
[19] T.A.B. Snijders, P.E. Pattison, G.L. Robins, M.S. Handcock, New specifications for exponential random graph models, Sociol. Methodol. 36 (2006) 99–153.
[20] B. Sturmfels, Solving Systems of Polynomial Equations, AMS, Providence, Rhode Island, 2002.